Opinnäytetyö (AMK)

Tietotekniikan Koulutusohjelma

NTIETS12H

2016

Joni-Matti Huotari

PUHEOPETUSPELIN KEHITYS UNITY-PELIMOOTTORIN JA LISÄOSAN AVULLA

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietotekniikan koulutusohjelma

2016 | 23

Joni-Matti Huotari

PUHEOPETUSPELIN KEHITYS UNITY-PELIMOOTTORIN JA LISÄOSAN AVULLA

Sanalanka on puheopetuksen tueksi tarkoitettu peli. Sanalanka on tarkoitus kääntää useille kielille. Käännösten mahdollistamista varten täytyi luoda tekninen perusta käännettyjä tekstejä varten.

Sanalangan hyödyntämää pelimoottoria Unityä ja sen lisäosaa Fungusta käyttäen luotiin ratkaisu, jonka avulla Sanalanka voidaan kääntää toisille kielille. Vaatimuksena oli, että uusien käännösten lisääminen olisi helppoa, joten tämä otettiin huomioon suunnittelussa ja toteutuksessa.

Lopputuloksena saatiin ratkaisu, jonka avulla Sanalankaan voidaan lisätä uusia kielivaihtoehtoja käyttämällä käännökset sisältävää taulukkomuotoista tiedostoa. Ratkaisun avulla tehtiin tulevaa käännöstyötä varten käännös pelin ensimmäiseen välianimaatioon.

Tehtyä työtä voidaan hyödyntää heti, kun Sanalankaan aletaan tehdä käännöksiä. Työn ansiosta uusia käännöksiä voi lisätä ilman laajaa Unityn osaamista.

ASIASANAT:

puheopetus, peliohjelmointi, kääntäminen

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information Technology

2016 | 23

Joni-Matti Huotari

SPEECH THERAPY GAME DEVELOPMENT WITH UNITY GAME ENGINE AND PLUGIN

Sanalanka is a videogame meant to be used in speech therapy. Sanalanka is going to be translated to multiple languages. To make these translations possible there was a need to create a technical foundation for the translated texts.

Using the game engine Unity which is used by Sanalanka, and its plugin Fungus, a solution was created for translating Sanalanka. Requirement for the translation system was that adding new language options has to be easy so this was taken in consideration in the planning and implementation.

The result was a solution which allows adding new language options to Sanalanka using a spreadsheet file containing the translations. The solution was used to create example translation to the game's first cutscene.

The thesis can be utilized when the translation of Sanalanka begins. Thanks to the thesis translations can be added without extensive knowledge of Unity.

KEYWORDS:

speech therapy, game programming, translating

SISÄLTÖ

KUVAT

TAULUKOT

SANASTO

Unity	monialustainen pelimoottori, johon on saatavissa ilmainen lisenssi
Fungus	Unityn ilmainen lisäosa tekstin ja valikoiden toteuttamista varten
Scene	Unityllä kehitettävä peli jaetaan pelitasoihin joita kutsutaan sceneiksi
CSV	yksinkertainen tiedostomuoto jossa tallennetaan tekstiä taulukkomuotoon, ja jota voidaan käsitellä taulukkolaskentaohjelmassa, Comma-Separated Value
JSON	tiedostomuoto, Javascript Object Notation
API	ohjelmistorajapinta, joka mahdollistaa eri ohjelmien välisen viestinnän, Application Programming Interface
Direct3D	ohjelmistorajapinta 3D-grafiikkaa varten
OpenGL	laitteistoriippumaton ohjelmistorajapinta grafiikkaa varten

1 JOHDANTO

Tämä opinnäytetyö käsittelee Unity-pelimoottoriin tehtyä laajennusta nimeltä Fungus, jota voidaan käyttää dialogin, valikoiden ja muun näytöllä näkyvän tekstin näyttämiseen. Osana opinnäytetyötä tehtiin Fungusta käyttävä järjestelmä helpottamaan ohjelmiston kääntämistä eri kielille. Työ oli osa projektia, jossa tehtiin puheopetukseen tarkoitettu opetuspeli nimeltä Sanalanka.

Opinnäytetyössä tutustutaan Unityyn ja Fungukseen siitä näkökulmasta, miten niitä projektissa käytettiin ja mitä vaihtoehtoisia ratkaisuja olisi voitu käyttää. Erityisesti keskitytään siihen, miten Funguksen käännöstoimintoja käytettiin Sanalangan lokalisoinniksi.

2 UNITY JA SEN LISÄOSAT

2.1 Unity

Pelimoottori tarjoaa pelinkehittäjille yleisiä toimintoja, joita pelit tarvitsevat. Näihin kuuluu esimerkiksi grafiikan piirtäminen, fysiikkamallinnus ja käyttäjän syötteen seuraaminen. Valmiin pelimoottorin käyttäminen nopeuttaa pelien kehittämistä. Tästä huolimatta monet yritykset tekevät itse pelimoottorinsa, koska monet saatavilla olevat pelimoottorit eivät sovellu suunnitellun kaltaisen pelin tekemiseen. (GameDesigning 2016)

Unity on Unity Technologiesin valmistama pelimoottori. Suurin osa pelien kannalta merkittävistä alustoista ovat yhteensopivia Unityn kanssa. Yhteensopivia alustoja ovat esimerkiksi Windows, OS X, Linux, Xbox, Playstation, Wii, Android, iOS ja Windows Phone. (Unity Technologies 2016c) Uusin versio on 28.7.2016 julkaistu 5.4.0f3. Unity on ilmainen käyttäjille, joiden vuosittainen liikevaihto on alle 100 000 Yhdysvaltojen dollaria, ja tämän ylittyessä on siirryttävä käyttämään kuukausimaksullista versiota. Maksullisissa versioissa on saatavilla lisätoimintoja, kuten suorituskyvyn mittaamistyökalu ja tuki useamman pelaajan moninpelille. Näiden kuukausimaksullisten lisenssien lisäksi on olemassa lisäosia myyvä Asset store, josta voi ostaa esimerkiksi 3D-malleja. (Unity Technologies 2016e)

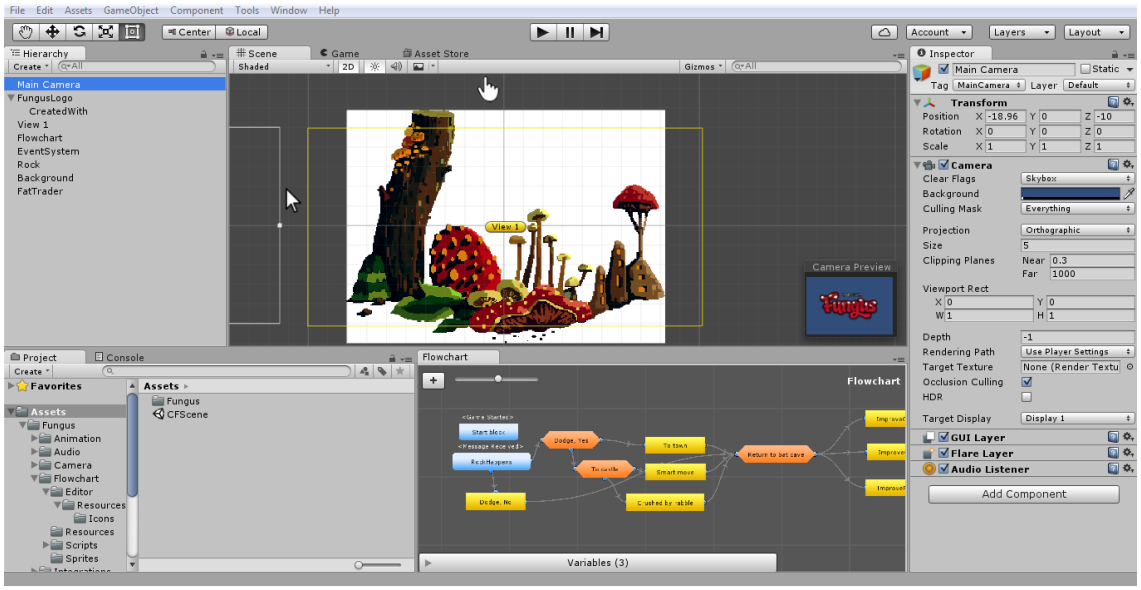
Ohjelmistorajapintoja (API) käytetään eri ohjelmien väliseen viestintään (Webopedia 2016). 3D-grafiikkaa varten oleva Direct3D on ohjelmistorajapinta (Microsoft 2016). Grafiikan näyttämiseen on olemassa myös laitteistoriippumaton OpenGL (OpenGL 2016).

Unityn tukemat ohjelmistorajapinnat ovat Direct3D (Windows ja Xbox), OpenGL (Mac, Linux ja Windows), OpenGL ES (Android ja iOS). Näiden lisäksi on olemassa yksityisomistukselliset ohjelmistorajapinnat konsoleille, kuten Playstationille ja Wiille. (Unity Technologies 2016c)

Unity tukee ohjelmointikieliä C# ja Javascript (Unity Technologies 2016d). Tuettuihin kieliin kuului myös Boo ennen versiota 5.0, jossa sen tuki lakkautettiin. Boo-skriptejä voi kuitenkin edelleen käyttää, mutta dokumentaatiota ei ole, ja painike, jolla uusi Boo-skripti luodaan, on poistettu. (Unity Technologies 2014)

Unityllä tehtyihin peleihin kuuluu esimerkiksi Kerbal Space Program, Hearthstone, Wasteland 2 ja Monument Valley.

Kuvassa 1 näkyy Unityn editori (Unity scene editor). Unityssä scene tarkoittaa pelikenttää, joita voi olla pelistä riippuen yksi tai useampia. Yksinkertaisen pelin voi tehdä yhtä otosta (sceneä) käyttäen mutta yleensä kannattaa käyttää useampaa. Otokseen on helppo tehdä yksinkertaisia muutoksia kuten pelihahmojen sijainnin muuttaminen tai taustan vaihtaminen.



Kuva 1. Unityn editori.

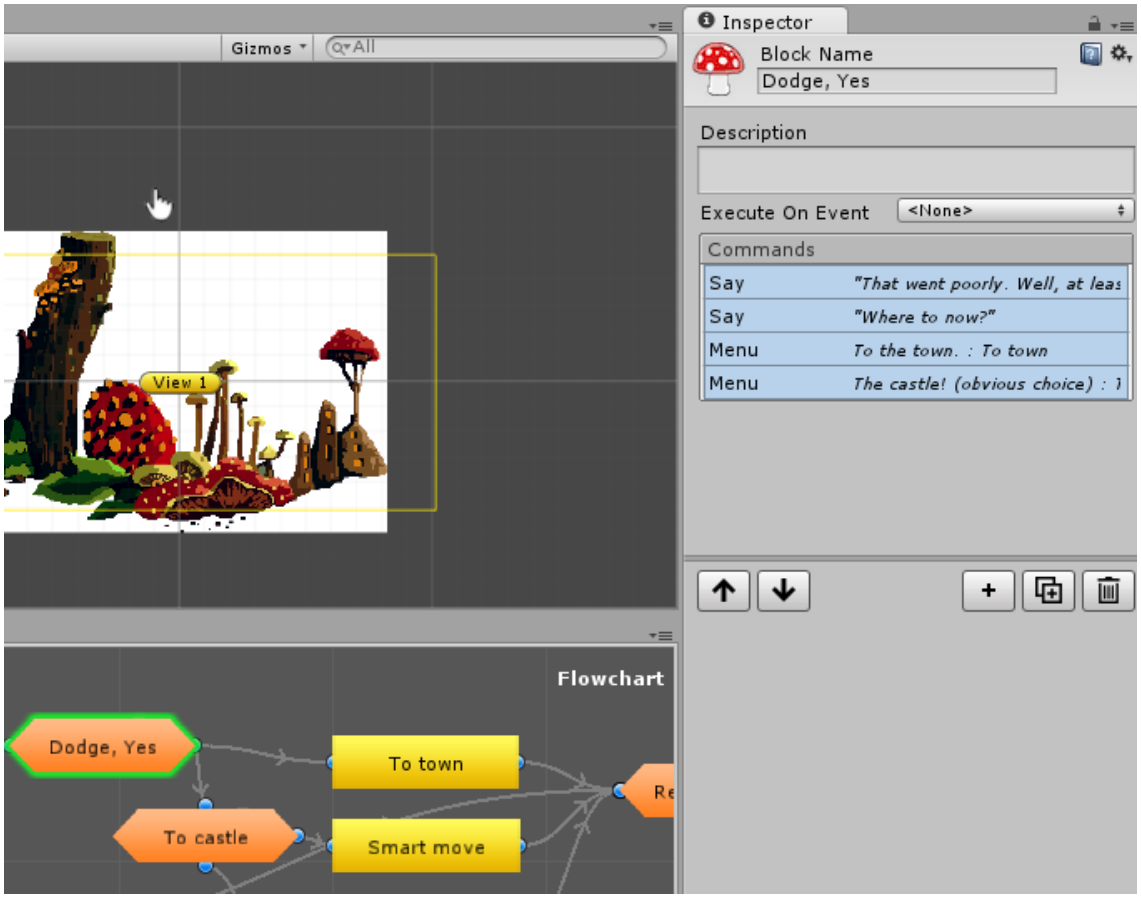
Unityn käynnistyessä on luotava uusi projekti tai valittava jo olemassa oleva projekti. Unity-projektiin kuuluvat kaikki tiedostot, joita käyttäjä tekee tai muuten hankkii peliään varten. Projektiin voi lisätä ilmaisia tai maksullisia skriptejä, grafiikoita ja muita tiedostoja.

Unityssä käytetään termiä GameObject eli peliobjekti. Peliobjekteilla on erilaisia ominaisuuksia, joista yleisin on sijainti. Objektit luodaan usein tyhjinä, jolloin niillä ei ole muita ominaisuuksia kuin sijainti. Tyhjään objektiin lisätään sitten tarvitut komponentit, kuten fysiikkamallinnus tai grafiikkaa. Objekteja, joissa ei ole komponenttia grafiikan näyttämiseen, ovat peliruudulla näkymättömiä. Jokainen otos tarvitsee objektin, johon on liitetty kamerakomponentti: se määrää, mitä peliruudulla näytetään.

2.2 Fungus

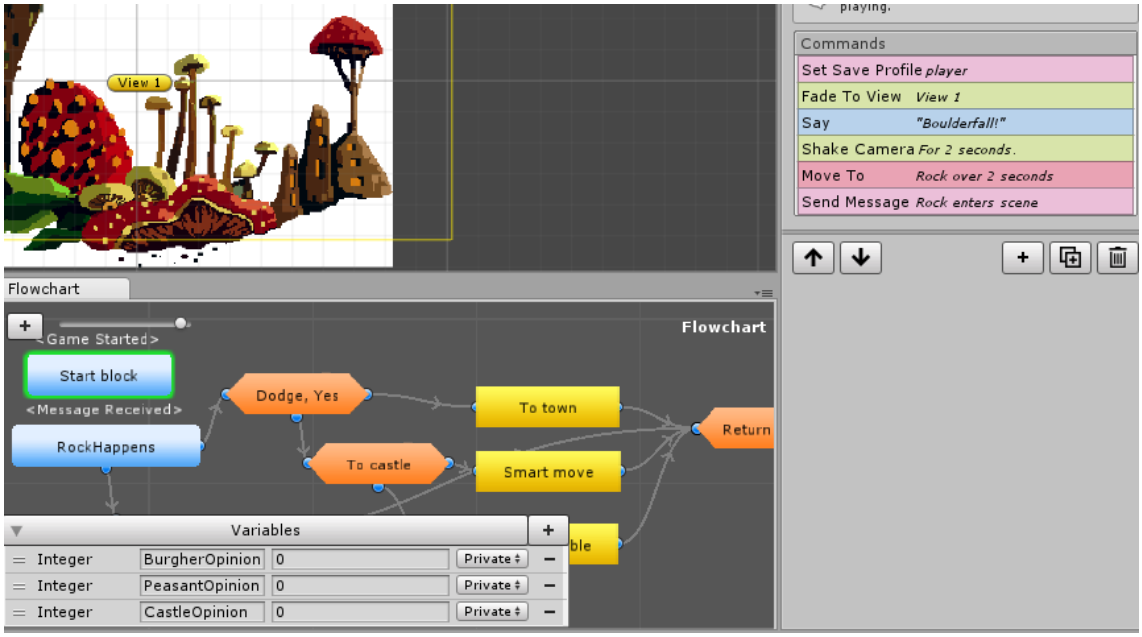
Fungus on ilmainen vapaan lähdekoodin lisäosa Unityyn. Fungus tuli saatavaksi Asset Storesta 4.6.2015 (Unity Technologies 2016a). Uusin versio on 25.7.2016 julkaistu 3.1.0. Funguksen keskeisimpiä toimintoja on dialogin ja valikoiden toteuttaminen peleissä. Funguksen avulla voi näyttää ja liikuttaa hahmojen potretteja ja soittaa äänitiedostoja. Hahmografiikoita voi liikuttamisen lisäksi skaalata ja litistää. Unityn animointityökalua Mecanimia voi ohjata Funguksen kautta tai sitten voi käyttää Funguksen mukana tulevaa animointilisäosaa nimeltä Spine. Pelilogiikkaa voi tehdä käyttäen Funguksen visuaalista ohjelmointia eli vuokaaviotoimintoa (flowchart). (Fungus 2016a)

Vuokaavio ohjaa funguksen näyttämää tekstiä. Kuutioihin (block) lisätään komentoja, jotka ohjaavat tapahtumien kulkua. Kuutioita on useampia, jotta tarina voi haarautua. Yksittäinen puhekupla näytetään Say-komennolla, valinnat toteutetaan Menu-komennolla. Kuvassa 2 näkyy, miten pelaajalle annetaan valikossa erilaisia vaihtoehtoja, jotka johtavat haarautuviin polkuihin.



Kuva 2. Funguksen vuokaavio.

Kuvassa 3 on kuvattu muuttujia joiden arvot vaihtuvat pelaajan valintojen mukaan. Myöhemmässä vaiheessa peliä muuttujien arvot vaikuttavat siihen mille polulle pelaaja päätyy.



Kuva 3. Fungus vuokaavio muuttujat ja kameran liikuttaminen.

Usean eri tyypin muuttujia voidaan lisätä vuokaavioihin ja niiden arvoja voi muuttaa vuokaaviokomennoilla. Kameraa ohjaavia komentoja on useita. Niiden kanssa on mahdollista liikuttaa kameraa useilla tavoilla, kuten tasaisella siirtämisellä tai heiluttamalla sitä. Kameran näyttämää kuvaa myös voi tummentaa tai kameraa voi siirtää kauemmas tai lähemmäs taustasta.

Say-komento on eräs Funguksen käytetyimmistä. Komentoa voi muokata monin tavoin tarpeiden mukaan. On mahdollista valita hahmo jonka potretti näytetään samalla tekstin kanssa. Muita mahdollisia asetuksia ovat esimerkiksi tekstin nopeus ja tekstin sijoittaminen aikaisempaan puhekuplaan uuden sijaan.

Menu-komennolla luodaan painike jossa lukeva lukeva teksti määritellään tekstikohdassa (text). Jokaiselle vaihtoehdolle on lisättävä oma komento. Kohde (target) määrää mistä kuutiosta valinnan jälkeen jatketaan.

Shake Camera -komento heiluttaa kameraa. Heilutuksen kesto päätetään ajastuskohdassa (time) ja heilutuksen määrä pitää ilmoittaa x ja y koordinaatteina,

esimerkiksi x = 0.3 ja y = 0.4. *Wait until finished* määrää, tehdäänkö heilutus loppuun ennen kuin vuokaavio jatkaa toteuttamaan seuraavaa komentoa.

Move To -komento siirtää valittua objektia. Siirrettävä objekti voidaan siirtää joko tiettyihin xyz-koordinaatteihin tai samaan paikkaan jonkin toisen objektin kanssa. Siirto tapahtuu kohtaan kesto (duration) asetetussa ajassa.

Set Variable -komennolla tehdään muutoksia vuokaaviossa oleviin muuttujiin. Variable kohtaan sijoitetaan muutettavana oleva muuttuja. Muuttujaan voidaan asettaa tietty arvo tai sitten voidaan tehdä yksinkertaisia laskutoimituksia joko toisen muuttujan kanssa tai komennossa ilmoitettavan luvun kanssa.

2.3 Muita vaihtoehtoja

Unityn lisäksi on olemassa useita muita pelimoottoreita joita voi käyttää maksutta. Epic Gamesin valmistama Unreal Engine on Unityn ohella eräs käytetyimmistä pelimoottoreista. Unreal Enginen ensimmäinen versio tehtiin alunperin Unreal peliä varten vuonna 1998. Uusin versio on Unreal Engine 4 ja se julkaistiin vuonna 2012. Unreal Enginestä tuli kaikille käyttäjille ilmainen vuonna 2015, tosin hyvin myyvästä pelistä täytyy maksaa rojalteja. Ohjelmointiin voi käyttää joko Unrealin visuaalista ohjelmointikieltä tai ohjelmointikieltä c++. (Unreal Engine 2016)

Unreal Engineä pidetään Unityä parempana graafisesti raskaisiin peleihin, mutta vaikeampikäyttöisenä. Unity on helppoutensa takia käytetyimpiä pelimoottoreita. Unityn helppouteen verrattuna Unreal Engineen vaikuttaa huomattavasti pelimoottoreiden käyttämät ohjelmointikielet. Unityn C# on helpompi oppia kuin Unreal Enginen käyttämä C++. (The Next Web 2016) Sanalangassa käytetään Unityä, koska Sanalanka ei tarvitse kehittyneitä grafiikoita ja koska Turku Game Labin henkilöstöllä on enemmän kokemusta Unityn käytöstä.

Unityn ja Unrealin lisäksi on muitakin ilmaisia pelimoottoreita. Saksalaisen Crytekin CryEngine soveltuu erityisesti 3D peleihin joissa tarvitaan hyvää suorituskykyä. CryEngineä myydään *maksa mitä haluat* -menetelmällä. Peleistä ei tarvitse maksaa rojalteja. (CryEngine 2016) Amazon.com teki CryEnginestä oman versionsa nimeltä Amazon Lumberyard. Lumberyardilla rakennettu peli voi hyödyntää Amazonin verkkopalveluita ja sitä voi suoratoistaa videopalvelu Twitchiin ilman erillistä ohjelmaa. Itse pelimoottori on ilmainen mutta verkkopalveluiden hyödyntämisestä täytyy maksaa.

(Amazon Lumberyard 2016) Taulukosta 1 nähdään pelimoottorien vertailu ohjelmointikielien, hinnan ja sopivimpien käyttökohteiden osalta.

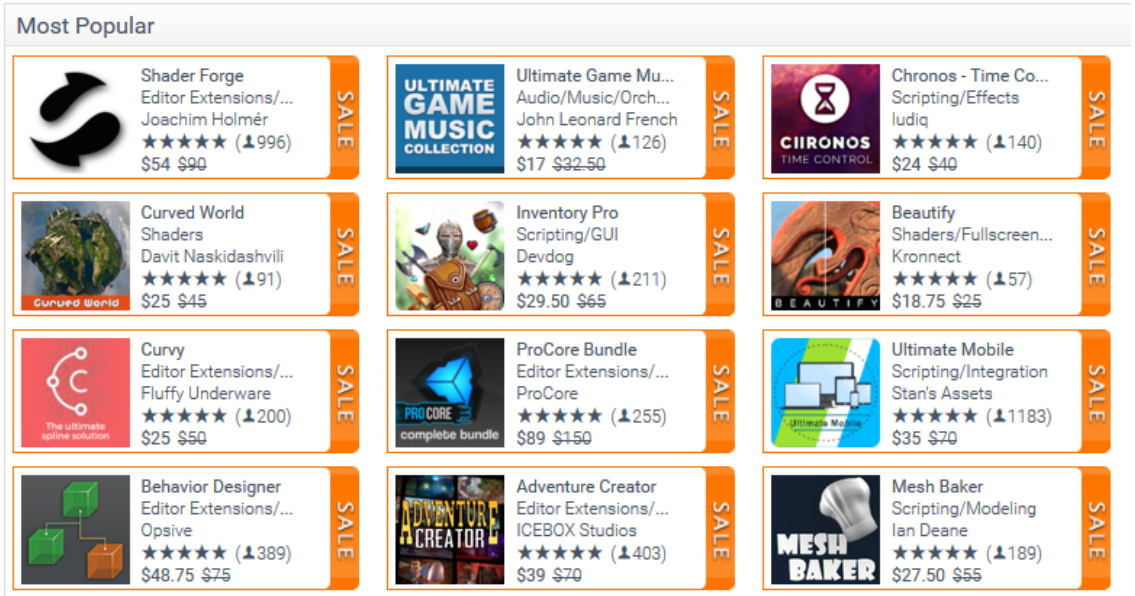
Taulukko 1. Pelimoottorien vertailu.

	Ohjelmointikieli	Hinta	Käyttökohde
Unity	C#, Javascript (eli Unityscript)	ilmainen, saatavilla maksullisia lisätoimintoja	yksinkertaiset pelit, mobiilipelit
Unreal Engine	C++	5 % rojalti myydystä tuotteesta	graafisesti raskaat pelit
CryEngine	C++	ilmainen	realistisia grafiikoita käyttävät pelit ja ulkotiloihin sijoittuvat pelit
Amazon Lumberyard	C++	ilmainen, nettipalveluista pitää maksaa	Amazonin verkkopalveluita tai Twitch-palvelua hyödyntävät pelit

Fungus on yksi monista Unityä varten olemassa olevista tekstin ja valikoiden näyttämiseen tarkoitetuista lisäosista. Monet lisäosat tarjoavat samankaltasia toimintoja kuin Fungus. Useimmat lisäosat joissa on paljon toimintoja ovat maksullisia, mutta on myös olemassa useampi täysin ilmainen vaihtoehto.

Lisäosia voidaan käyttää nopeuttamaan pelin kehitystä. Lopputuloksen kannalta ei ole juuri merkitystä tekikö pelin kehittäjäryhmä itse kaikki 3D-mallit tai skriptit. Lisäosia voi myös käyttää ideoiden nopeaan toteuttamiseen ja testaamiseen. Tällöin on mahdollista testata ideoita ilman, että täytyy käyttää paljon aikaa esimerkiksi uusien hahmojen luomiseen. Joskus kehittäjät käyttävät lisäosia täyttämään jonkin puutteen omissa taidoissaan. Esimerkiksi graafikko ilman ohjelmointitaitoja voi hyödyntää muiden tekemiä skriptejä, tehdäkseen pelimekaniikoiltaan yksinkertaisen, mutta graafiselta suunnittelultaan ja toteutukseltaan laadukkaan pelin. Fungusta käyttämällä voi heti

aloittaa välianimaatioiden toteuttamisen ilman tarvetta koodata tyhjästä niiden vaatimia skriptejä. Kuvassa 4 on joitakin Asset Storesta saatavilla olevia maksullisia lisäosia.



Kuva 4. Asset Store.

Ilmaisia Funguksen kaltaisia lisäosia ovat esimerkiksi Narrate ja Parley Free. Narrate mahdollistaa dialogin näyttämisen ruudulla. Narrate on yksinkertainen eikä se tue haarautuvia keskusteluja. Parley Free Dialogue & Quest Engine on monipuolisempi, se tukee haarautuvia keskusteluja kuten Funguskin.

Parley vaatii käyttäjältä Java ohjelmointikielen asennuksen mikä vaikeuttaa Parleyn käyttöä. Sovellus joka hyödyntää Parleyn ominaisuuksia ei toimi ellei käyttäjällä ole yhteensopivaa versiota Javasta. Parley Free on ilmaisversio jossa ei ole kaikkia ominaisuuksia mitä maksullisesta versiosta löytyy.

Maksullisia lisäosia ovat Chat UI & Speech Bubbles System, Dialogue and Conversation System, Dialogue System for Unity ja Visual Novel Engine & 2d Cutscene Engine. Chat UI on tarkoitettu erityisesti useiden pelaajien väliseen keskusteluun joten se ei ole sopivin Sanalankaa varten. Siinä ei myöskään ole tukea lokalisoinnille. Chat UI & Speech Bubbles System maksaa 25 dollaria.

Dialogue and Conversation System on monipuolinen ja se tukee esimerkiksi äänitiedostojen soittamista ja lokalisointia. Siinä on myös toimintoja kameran liikuttelemiselle. Dialogue and Conversation System maksaa 40 dollaria.

Dialogue System for Unity sisältää paljon toimintoja kuten monipuoliset vaihtoehdot keskusteluiden tekemiseen, tuen lokalisoinnille, hahmojen välisten suhteiden seuraamisen ja pelin tallentamisen sekä lataamisen. Dialogue System for Unity maksaa 65 dollaria.

Visual Novel Engine & 2d Cutscene Engine on tarkoitettu erityisesti 2-ulotteisia pelejä varten joissa on paljon tekstiä ja yksinkertaiset pelimekaniikat. Se sisältää toiminnallisuuden muutaman hengen välisiin keskusteluihin ja hahmografiikoiden liikuttelemiseen, mutta siinä ei ole juuri muita toimintoja. Visual Novel Engine maksaa 5 dollaria. (Unity Technologies 2016b)

3 SANALANKA

3.1 Sanalanka yleisesti

Sanalanka on puheopetuksen tueksi tarkoitettu opetuspeli. Sanalankaa kehittää Turku Game Lab ja tämä työ tehtiin sopimuksessa Turku Game Labin kanssa. Turku Game Lab on Turun yliopiston ja Turun ammattikorkeakoulun yhteinen hanke jossa toteutetaan peliprojekteja. Sanalanka on suunniteltu erityisesti taulutietokoneita varten. Sanalanka koostuu useasta minipelistä joissa opitaan sanojen oikea ääntäminen. Erilaisia minipelejä on 6 ja peli on jaettu 8 alueeseen, joille on Unityssä oma otos. Jokaisella alueella suoritetaan kaikki minipelit. Minipelit ovat samanlaisia joka kerralla muuten paitsi harjoiteltavan sanaston kohdalta. Suunniteltua on että Sanalankaa voisi pelata suomeksi ja ruotsiksi. Tulevaisuudessa kieliä saatetaan lisätä enemmän joten peli toteutettiin niin että useiden eri käännösten lisääminen myöhemmin olisi helppoa.

3.2 Fungus Sanalangassa

Sanalangassa käytetään Fungusta erityisesti tarinaa edistävien välianimaatoiden toteutukseen. Pelin alettua näytetään pelaajalle alkuanimaatio, mikä selittää pelin tapahtumien kontekstin. Välianimaatioita näytetään myös pelaajan saapuessa uuteen pelialueeseen.

Välianimaatioissa Fungusta käytetään tekstin näyttämiseen, puheen ja muiden äänien soittamiseen, hahmojen liikuttamiseen ja kameran hallintaan. Riippuen valitusta kielestä fungus käyttää joko suomen tai ruotsin kielistä tekstiä ja puhetta. Hahmojen animaatiot toteutataan joko Funguksen toiminnoilla, tai sitten Fungusta käytetään vain kutsumaan metodi joka sitten toteuttaa animaation. Fungusta ei käytetä kaikkeen, esimerkiksi pelialueen vahtamiseen käytettävä kartta ja minipelit ovat toteutettu ilman sitä. Kuvassa 5 oleva tekstitys on tehty Fungusta käyttäen.

Kuva 5. Fungus tekstitys välianimaatiossa.

Sanalangassa hahmografiikoita liikutetaan käyttäen move to -komentoa. Move to -komento vaatii tiedon siitä mitä halutaan siirtää, kuinka paljon ja kuinka nopeasti. Hahmojen ja esineiden animaatiot tapahtuvat käyttäen Funguksen toimintoa jolla voi kutsua itse tehtyjä metodeita. Välianimaation loppuessa pelaajalle täytyy palauttaa kyky ohjata peliä ja se tehdään animaatioiden tavoin kutsumalla metodi mikä antaa kontrollin takaisin pelaajalle.

Sanalangan välianimaatoissa teksti ja puhe tehtiin ensin käyttäen Funguksen vuokaavioon lisättävällä say-toiminnolla. Say-komentoon yhdistettiin äänitiedosto joka soitettiin samaan aikaan tekstin kanssa. Tämä toteutus ei mahdollistanut tekstin tai puheen kääntämistä toiselle kielelle, minkä vuoksi otettiin käyttöön Funguksen toiminto jossa teksti ja siihen liittyvä ääni tulee taulukkomuotoisesta CSV-tiedostosta.

Fungus sisältää tuen tekstin kääntämiselle. Vuokaavion voi muuttaa automaattisesti kuvan 6 kaltaiseen CSV-muotoon. CSV-tiedostoa voi muokata taulukkolaskentaohjelmien kanssa (CSV 2016). Funguksen luomaan tiedostoon tulee jokainen muutetun vuokaavion say-komennon sisältämä teksti omalle rivilleen. Jokainen näistä teksteistä liittyy avainriviin mikä kertoo missä kohdassa teksti näytetään.

	A	B	C	
1	Key	Description	Standard	SWE
2	SAY.Flowchart.21.		{audio=AlusRikki}Vaikka tÃ¤mÃ	{audio=AlusRikki2}Även om detta rymdskepp
3	SAY.Flowchart.52.		MMMMMM	MMMMMM
4	SAY.Flowchart.22.		{audio=Kuka}Kuka tulee avaruus	{audio=Kuka2}Vem blir rymdskepp?{w=1.2} Ä
5	SAY.Flowchart.13.Miima		UÃ¤hÃ¤Ã¤hÃ¤Ã¤	UÃ¤hÃ¤Ã¤hÃ¤Ã¤
6	SAY.Flowchart.14.Ã„iti		Shh, shh	Shh, shh

Kuva 6. Käännökseen käytettävä CSV-tiedosto.

Tiedostoon voi lisätä minkä tahansa määrän käännöksiä. Tyhjän sarakkeen ylimpään riviin tulee sijoittaa avain jolla viitataan sillä sarakkeella olevan käännöksen kieleen. Yläpuolella olevassa esimerkissä on lisätty avain SWE ja sen alle on tehty itse käännös.

CSV-tiedoston käyttämistä varten on otokseen lisättävä lokalisaatio (localization) tyypin objekti, mihin tulee lisätä haluttu tiedosto. Unity käyttää tiedostosta nimeä localization file. Kun localization file on lisätty se korvaa vuokaavion komentojen sisällön vastaavalla rivillä olevalla tekstillä. Sarake, josta teksti otetaan, määräytyy päävalikossa valitun kielen mukaan.

Äänien lokalisointi tapahtuu lisäämällä käännöstiedostoon viittaus otoksessa olevaan objektiin johon on liitetty haluttu äänitiedosto. Viittaus on muotoa {audio=x}, missä x on objektin nimi. Kun halutaan lisätä uusia ääniä täytyy jokaiselle uudelle äänelle tehdä oma objekti. Tällöin uuden käännöksen lisääminen vaatii taulukkotiedoston muuttamisen lisäksi otoksessa tapahtuvia muutoksia. Tämä eroaa siitä miten ääniä lisätään käyttäessä vuokaavion say-komentoa. Tarkasteltaessa say-komentoa voidaan siihen helposti lisätä mikä tahansa projektissa oleva äänitiedosto. Sen sijaan CSV-tiedostoa käyttäessä jokainen ääni tarvitsee otoksessa olevan objektin.

3.3 Ääniobjektien tekemien skriptin avulla

On mahdollista että jossain vaiheessa käännöksiä tulee tekemään joku jolla ei ole kokemusta Unitystä eikä Funguksesta. Tämän vuoksi on tärkeää että käännöstyötä helpotetaan. Ääniä sisältävien objektien tekeminen yksi kerrallaan on myös aikaavievää ja virhealtista. Ratkaisuna tähän tehtiin skripti mikä tekee otokseen automaattisesti objektin jossa on haluttu äänitiedosto.

Kuvassa 7 kuvatun koodin ansiosta uusien äänien lisääminen on helppoa. Uuden äänen lisääminen vaatii vain sen että haluttu äänitiedosto vedetään niille varattuun clips-

taulukkoon. Tämän jälkeen ääntä voi käyttää CSV-tiedostossa viittaamalla siihen sen omalla nimellä.

```
public GameObject AudioLocalization;
public AudioClip[] clips;
void Start ()
{


    string lang = LanguageManager.Instance.CurLanguage.ToString();

    GameObject go = GameObject.Find("Localization");
    if(go != null)
    {
        go.GetComponent<Localization>().SetActiveLanguage(lang);
    }

    if(clips.Length > 0) {

        for (int i = 0; i < clips.Length; i++)
        {
            GameObject soundfile;
            soundfile = new GameObject(clips[i].name); //loops for current audioclip and
            soundfile.transform.parent = AudioLocalization.transform; //attach to parent
            soundfile.AddComponent(typeof(AudioSource));
            AudioSource audio = soundfile.GetComponent<AudioSource>();
            audio.clip = clips[i]; //here it loops the list for current audioclip
        }
}
```

Kuva 7. Skripti joka luo objektit äänitiedostoille.

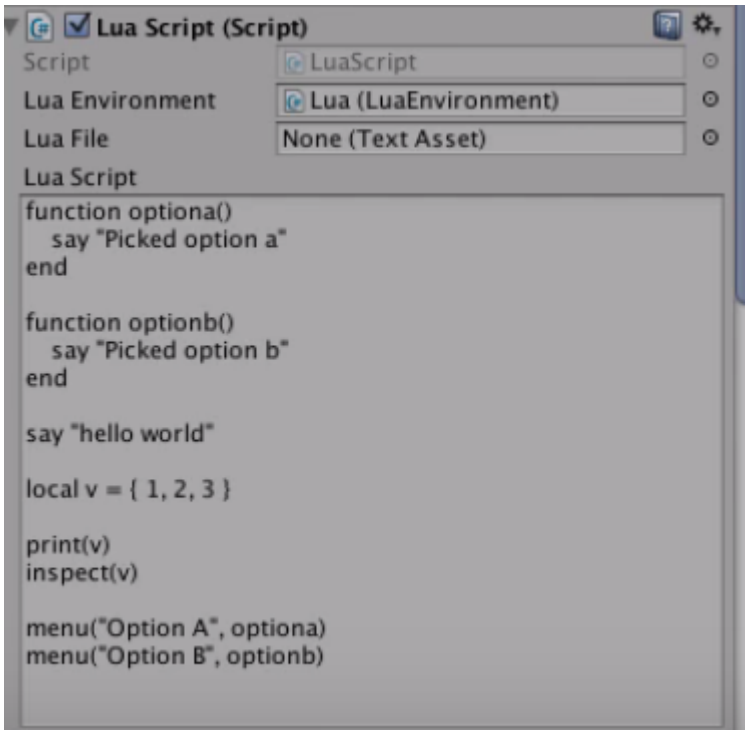
Skripti selvittää päävalikossa valitun kielen ja sitten etsii otoksesta lokalisaatio objektin johon valittu kieli sijoitetaan. Tämän jälkeen skripti luo jokaiselle taulukossa olevalle äänitiedostolle objektin, nimeää objektit äänitiedostojen mukaan, siirtää objektit hierarkiassa ennalta tehdyn audiolokalisaatio objektin alle ja lisää äänitiedostot oikeisiin objekteihin.

Ääniobjektien lisäämisen olisi voinut tehdä myös muilla tavoilla. Ensimmäinen suunnitelma oli että jokaisen otoksen audiolokalisaatio objektiin kiinnitetään pelin käynnistyessä objekteja ääniä varten sen kielen mukaan mikä päävalikossa valittiin. Kun päävalikossa valitaan kieleksi suomi lisättäisiin otokseen objektit vain niille äänille joita tarvitaan suomenkielisessä Sanalangassa. Tehtyihin objekteihin lisättäisiin sitten vastaava ääni. Tämän olisi voinut toteuttaa käyttäen jonkinlaista taulukkoa josta voitaisiin valitusta kielestä riippuen lukea mitä ääniä käytetään. Tämä toteutus oli tarpeettoman hankala toteuttaa koska riippuen valitusta kielestä täytyisi otokseen lisätä eri ääniä. Lisäksi monimutkaisempi toteutus aiheuttaa todennäköisemmin virheitä.

3.4 Fungus ja Lua

Lua on skriptikieli jota Fungus tukee. Funguksen kanssa toimivan Luan nimi on FungusLua. Luan kanssa voi muokata Unityn objekteja ja sitä voi käyttää yhdessä Funguksen vuokaavioiden kanssa. Vuokaavioiden komentoja kuten say ja menu voi käyttää myös Luan kanssa. (Fungus 2016c)

Lua-skriptin voi kirjoittaa editorissa lisäämällä otokseen Lua-objektin. (Tools > Fungus > Create > Lua) Lua-skriptin voi kirjoittaa editorissa sille varattuun tilaan tai txt-tyyppiseen tiedostoon (Lua File). Kuvan 8 esimerkissä oleva skripti näyttää ensin tekstin *hello world* ja sen jälkeen näyttää valikon jossa on kaksi vaihtoehtoa. Saman toiminnallisuuden voi siis tehdä joko käyttäen Funguksen vuokaavio toimintoa tai Lua-skriptejä käyttäen. Lua-skriptejä voi myös kutsua vuokaavioiden avulla. (Fungus 2016c)



Kuva 8. Lua skripti editorissa. (Fungus 2016d)

Lualla näytettävän tekstin voi lokalisoida käyttäen JSON-tyyppistä tiedostoa johon käännökset on kirjattu seuraavan esimerkin mukaisesti. (JSON 2016)

```
{
    "hello_world" : {
        "en" : "Hello world!",
        "fr" : "Bonjour le monde!",
```

```
        "de" : "Hallo Welt!"
    },
    "goodbye_world" : {
        "en" : "Goodbye world!",
        "fr" : "Au revoir monde!",
        "de" : "Auf Wiedersehen Welt!"
    }
}
```

(Fungus 2016b)

Kielen voi vaihtaa Lua-objektin asetuksista tai funktiolla setlanguage (languagecode). Kun kieli on asetettu voidaan haluttu käännös saada näkyviin skriptillä say(”{$hello_world}”). (Fungus 2016c)

Sanalangassa välianimaatiot on luotu vuokaavioiden eikä Lua-skriptien kanssa joten JSON-tiedostoja ei ole käytetty lokalisointiin.

4 LOPUKSI

Tehtävän tavoitteena oli luoda ohjelmistoarkkitehtuuri Sanalangan tulevaa käännöstyötä varten. Tavoite saavutettiin ajallaan ja Sanalanka tullaan kääntämään muille kielille hyödyntäen tehtyä selvitystä ja toteutusta. Lokalisointi voidaan tehdä CSV-tiedostoja käyttäen varsin helposti ilman suurta Unityn osaamista. Lokalisointia helpottamaan luotiin skripti, joka automatisoi peliobjektien luomisen äänitiedostoille. Selvisi, että Fungus soveltuu oikein hyvin Sanalangan 2D-animaatioiden toteuttamiseen ja lokalisointiin.

Unityn kanssa työskentelyssä ei ilmennyt ongelmia. Asiaan vaikutti se, että Unity ja C# olivat jo entuudestaan tuttuja. Vaikka Fungus oli uusi tuttavuus, sen käytössä ei juuri tullut ongelmia. Funguksen dokumentaatiosta oli paljon hyötyä.

Funguksen käytöstä saatua tietoa voi hyödyntää muissa projekteissa, joissa Fungusta käytetään samankaltaisilla tavoilla kuin Sanalangassa. FungusLuan käyttöä voisi tutkia lisää. Sanalangassa Luaa ei ole käytetty, mutta se voisi olla hyödyllinen jossakin toisessa projektissa.

LÄHTEET

Amazon Lumberyard 2016. Wikipedia. Viitattu 15.9.2016 https://en.wikipedia.org/wiki/Amazon_Lumberyard

CryEngine 2016. Wikipedia. Viitattu 11.9.2016 https://en.wikipedia.org/wiki/CryEngine

CSV 2016. Wikipedia. Viitattu 25.8.2016 https://fi.wikipedia.org/wiki/CSV

Fungus 2016a. Everyone loves a good story. Viitattu 17.8.2016 http://fungusgames.com/

Fungus 2016b. String table. Viitattu 16.9.2016 http://snozbot.github.io/fungus_lua/string_table/index.html

Fungus 2016c. What is FungusLua? Viitattu 11.8.2016 http://snozbot.github.io/fungus_lua/index.html

Fungus 2016d. FungusLua: Lua scripting support for Unity and Fungus. 5:30. Viitattu 16.9.2016 https://www.youtube.com/watch?v=M_Oo9FpVTos

GameDesigning 2016. What are game engines? Viitattu 16.9.2016 http://www.gamedesigning.org/career/video-game-engines/

JSON 2016. Wikipedia. Viitattu 11.9.2016 https://fi.wikipedia.org/wiki/JSON

Microsoft 2016. Direct3D Graphics. Viitattu 13.9.2016 https://msdn.microsoft.com/en-us/library/windows/desktop/bb153256(v=vs.85).aspx

OpenGL 2016. About. Viitattu 13.9.2016 https://www.opengl.org/about/

The Next Web. This engine is dominating the gaming industry right now. Viitattu 9.9.2016 http://thenextweb.com/insider/2016/03/24/engine-dominating-gaming-industry-right-now/#gref

Unity Technologies 2014. Documentation, Unity scripting languages and you. Viitattu 16.9.2016 https://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/

Unity Technologies 2016a. Asset store – Fungus. Viitattu 15.9.2016 https://www.assetstore.unity3d.com/en/#!/content/34184

Unity Technologies 2016b. Asset store. Viitattu 16.8.2016 https://www.assetstore.unity3d.com/en/

Unity Technologies 2016c. Build once deploy anywhere. Viitattu 16.8.2016 http://unity3d.com/unity/multiplatform/

Unity Technologies 2016d. Documentation. Viitattu 7.9.2016 http://docs.unity3d.com/ScriptReference/

Unity Technologies 2016e. New products and prices. Viitattu 17.8.2016 https://blogs.unity3d.com/2016/05/31/new-products-and-prices/

Unreal Engine 2016. Wikipedia. Viitattu 9.9.2016 https://en.wikipedia.org/wiki/Unreal_Engine

Webopedia 2016. API. Viitattu 13.9.2016 http://www.webopedia.com/TERM/A/API.html